The LogicalMulticriteriaSort ThinkLet: Logical Navigation for Fair and Fast Convergence in Multicriteria Group Decision Making

Mireille Ducassé, Peggy Cellier

Group Decision and Negotiation – GDN 2012 – Adial Teixeira de Almeida, Danielle Costa Morais and Suzana de França Dantas Daher, eds. 20-24 May 2012, Recife, Brazil. ISBN 978-85-415-0036-4

The LogicalMulticriteriaSort ThinkLet: Logical Navigation for Fair and Fast Convergence in Multicriteria Group Decision Making

Mireille Ducassé[1] and Peggy Cellier [1]

[1] IRISA-INSA de Rennes, Campus Universitaire de Beaulieu, 35042 Rennes, France

Mireille.Ducasse@irisa.fr, Peggy.Cellier@irisa.fr

Abstract: Information overload is a key issue in group decision. A heuristics, called "*take-the-best*", has been shown useful to face multicriteria decisions while reducing information overload: when making decisions people often take criteria in a predefined order, the first criterion which discriminates the alternatives at stake is used to make the decision. In order to rationalize group work, Briggs and de Vreede have proposed collaboration design patterns, called thinkLets. This article presents the LogicalMulticriteriaSort thinkLet that can be seen as a generalization of the take-the-best heuristics. It also proposes to consider criteria one at the time but once a criterion has been found discriminating it is kept in a record, and the process is iterated. The thinkLet is supported by a GDSS, based on Logical Information Systems, which gives an instantaneous feedback of each micro decision and keeps tracks of all of the decisions taken so far. The LogicalMulticriteriaSort thinkLet guarantees more fairness and speed than the ChauffeurSort thinkLet. It also avoids the need to give artificial values and weights to the criteria as opposed to the Multicriteria thinkLet. A successful test case is reported.

Keywords: Multicriteria decision, Logical Information Systems, Formal Concept analysis

1 Introduction

At the workplace, collaboration is a key issue, in particular, to make sensitive decisions. Important advantages of a group decision are that the group can share a better understanding of the situation than a single person and that it can be easier to collectively endorse a decision. Experience shows, however, that collaborative work is not always satisfactorily organized. Collaboration design patterns, called thinkLets, have been proposed (Briggs and de Vreede, 2009) to help the person, called the facilitator, in charge of helping a group achieve a common task. "*A thinklet provides a transferable, reusable and predictable building block for the design of a collaboration process*." A major problem faced by people is information overload. ThinkLets for convergence are especially needed in order to reduce the cognitive load (Davis et al. , 2007). As stated by (Lewis, 2010), "*Perhaps the greatest challenge a group will face is how to take a raw list of ideas and discuss, edit, and organize these ideas to create a coherent result*." Consistently, (Vogel and Coombes, 2010) state that, from a cognitive point of view, it is much more difficult for a group to organize ideas than to generate them. In order to minimize the effects of information overload, people tend to employ conscious or even unconscious heuristics (Vogel and Coombes, 2010). The "*gaze heuristics*", reported in (Marewski et al., 2010), is exemplary. In order to catch a ball high up in the air, a player fixates it, starts running and keeps the angle of gaze constant. He does not beforehand calculate a complex differential equation but he will be at the proper place to catch the ball.

A second important issue is the use of multiple criteria. The results of (Vogel and Coombes, 2010) show that "*groups selecting ideas from a multiple criteria task formulation will converge better than groups working on a single criterion formulation*". Another of the heuristics described in (Marewski et al., 2010), called "*take-the-best*", allows to face multicriteria decisions while reducing information overload: when making decisions people often take criteria in a predefined order, the first criterion which discriminates the alternatives at stake is used to make the decision. This heuristics has been shown to be more effective than multiple regression while considering less criteria in a number of cases. Not all criteria are, thus, relevant for a given decision. It is even crucial to discriminate against relevant and irrelevant criteria because, as stated by (Shanteau, 1992), irrelevant criteria can inappropriately influence the judgment of people whatever their expertise level.

The contribution of this article is to propose the LogicalMulticriteriaSort thinkLet with a supporting tool which could be an asset if integrated into existing toolboxes, such as for example the set of simple editors to help non professional facilitators proposed by (Briggs et al. , 2010) or the toolkit for GDSS facilitators proposed by (Adla et al., 2011). The LogicalMulticriteriaSort thinkLet gives a procedure to logically sort candidates into categories according to multiple criteria. It can be seen as a generalization of take-the- best and an implementation of the gaze heuristics. Following take-the-best, it proposes to consider criteria one at the time. There is no predefined order, participants put forward the ones they find the most relevant at a given time. We enhance take-the-best by using a GDSS, based on Logical Information Systems (LIS) (Ferré and Ridoux, 2004). It firstly enables participants to easily navigate in the data. It secondly gives an instantaneous feedback of each micro decision and it thirdly builds a shared knowledge by keeping tracks of all of the decisions taken so far. It thus provides support for the gaze heuristic with no cognitive overload. The LogicalMulticriteriaSort thinkLet has been tested on the debriefing of an academic year validation jury whose results had been controversial. The test case participants were positive about the process and the main result was that they all agreed to use the LogicalMulticriteriaSort thinkLet and the supporting tool for the forthcoming jury at the same level.

The LogicalMulticriteriaSort thinkLet is related to two thinkLets of (Briggs and de Vreede, 2009): ChauffeurSort and Multicriteria. A detailed discussion of these two thinkLets can be found in Section 3. In summary, compared to the ChauffeurSort thinkLet, or Organize of *Meetingworks*™ (Lewis, 2010), the major difference is that, with LogicalMulticriteriaSort, the discussion is led by the criteria and not by the candidates. It ensures both fairness and speed. Indeed, with our approach, it is guaranteed that all the candidates will have been considered along the discussed criteria. Furthermore, the meeting can stop after any criterion analysis if the group decides so. The Multicriteria thinkLet of (Briggs and de Vreede, 2009) also addresses the analysis of problems defined by multiple criteria. The values for the criteria, however, are necessarily numbers; weights are attached to each criterion and a weighted sum is computed. Producing numerical criteria weights is tedious, arbitrary and can lead to dubious results. With our approach, the values can be numerical or symbolic; their analysis is logical. Participants only have to decide if a criterion is relevant or not and define thresholds for the values of the relevant criteria. Mesta (Hiltunen et al., 2009) is a multicriteria decision support tool. For each criterion, participants are asked to propose acceptable thresholds. The graphical display and handling of the thresholds has been proved useful. The approach can be implemented on top of LIS tools. An interface `a la Mesta for the simple cases would be helpful. Our tool, nevertheless, enables to state logical queries that can be much more sophisticated.

In the following, Section 2 briefly introduces Logical Information Systems. Section 3 describes in detail the LogicalMulticriteriaSort thinkLet following the schema of (Briggs and de Vreede, 2009). Section 4 reports about the jury test case, and presents the users' feedbacks collected in two questionnaires, one before and one after the meeting.

2 Logical Information Systems

Logical Information Systems (LIS) (Ferré and Ridoux, 2004) belong to a paradigm of information retrieval that combines querying and navigation. LIS are formally based on Logical Concept Analysis (LCA) (Ferré and Ridoux, 2000), a logical generalization of a mathematical theory, Formal Concept Analysis (FCA) (Ganter and Wille, 1999). In LCA, logical formulas are used to describe objects, as opposed to FCA where only simple symbolic criteria can be used. One advantage of logical formulas is, for instance, to permit numerical and symbolic criteria to be combined. From the descriptions of objects, a data structure, called *concept lattice*, is computed. The concept lattice partially orders both objects and criteria. It serves as the navigation structure. Logical formulas are also used to represent queries and navigation links in the lattice. In the cases addressed by the LogicalMulticriteriaSort thinkLet, the data are so dense that the current FCA tools (see for example (Tilley, 2004)) that graphically display the concept lattices cannot be used. Indeed, the global concept lattice is too large to be managed by hand. On the opposite, and as illustrated by the screen copies in the following, the local views of LIS tools enable users to grasp and manage the relevant information. Local views are obtained by querying the system. There exist three ways to define a query: by formula edition, by navigation (selecting criteria in order to modify the query) or by examples. Query definition is illustrated in Section 4. It is one of the strengths of FCA and LCA systems to be able to focus on the criteria and the objects are then logically sorted. Another important user action provided by LIS tools is annotation. Annotations, navigation and querying can be performed in the same interface. In the following the examples are given using the Camelis tool [1].

3 The LogicalMulticriteriaSort ThinkLet

This section describes the LogicalMulticriteriaSort thinkLet following the schema of (Briggs and de Vreede, 2009). We have added a *Tool and Manpower* field. ThinkLets have been partitioned into 6 pattern categories (Kolfschoten et al., 2010): Generate, Reduce, Clarify, Organize, Evaluate and Consensus Building. The main related pattern and subpattern of LogicalMulticriteriaSort is ***Organize/Categorizing***. People speak of Convergence for the Reduce and Clarify patterns. LogicalMulticriteriaSort can also be considered as a convergence thinkLet because the discussions related to the relevance of criteria and the traces kept by the LIS tool highly contribute to build an explicit shared understanding of the situation. Furthermore, sorting into categories and filtering along the criteria contribute to the reduction aspects. Related thinkLets are discussed in the *Insights on LogicalMulticriteriaSort* field. The descriptions are illustrated using the test case detailed in Section 4, namely a jury to validate students' year, at a technical university. In the following, phrases in italic are taken from thinkLet descriptions of (Briggs and de Vreede, 2009). In that case the name of the thinkLet is given in between parentheses.

Choose this thinkLet...

... when you want to assure that the placement of every item/candidate in a category is carefully considered by the team (ChauffeurSort)

... to evaluate a list of items/candidates against multiple criteria (Multicriteria)

... when it is crucial that the decision process is as fair as possible

... when the important criteria for the decision are not necessarily known

... when creating a shared understanding of the discriminating criteria is crucial

... when the group will have to endorse a sensitive decision

... when meeting time is of the essence

Do not choose this thinkLet...

... when nobody has time to prepare the table associating criteria values to candidates

... if the group has not been prepared to make a decision based on multiple criteria

Overview

The group decides which of the criteria are discriminating for the sort, and what are the required values for those criteria. Candidate sorting is a logical consequence of these decisions. For example, for a jury, the group can decide that in order to pass a student must have had a given grade above a given threshold.

Inputs

1. A set of candidates, for example students for a jury.
2. A set of categories in which the candidates should be sorted. For a jury the categories can be "automatically pass", "let through by jury" and "fail".
3. A set of criteria. For a jury the criteria can be the average grades on different modules, how many credits students have validated, whether they fulfilled the English test requirement.
4. Values of these criteria for most of the candidates. Note that it is not mandatory that all candidates have all criteria filled in.
5. (optional) A set of rules, mandatory or revisable, that automatically sort subsets of the candidates. Mandatory jury rules tell which of the students automatically validate the year according to the

[1] see http://www.irisa.fr/LIS/ferre/camelis/

rulings of the university. An example of revisable jury rule is that students with a scientific grade average below a given threshold fail. The group is free to revise that threshold, or even remove the rule.

Outputs

1. Candidates sorted by category.
2. A set of criteria important for that particular decision.
3. A set of rules to explain the sorting. These rules can be used as germs for the next meetings of the same kind.

How to use LogicalMulticriteriaSort

Setup

1. The chauffeur, namely the person in charge of the technical aspects of the hardware and software (Lewis, 2010), collects the data to build the context, basically a table with a line per candidate, filling the criteria slots whenever it is relevant. Note that there are a number of cases where the prerequisite data is not an extra burden because it is done anyway. For example, for juries, grades and additional information must be collected whatever thinkLet is used.
2. The chauffeur or the facilitator integrates the data into the LIS tool.
3. The facilitator investigates the data in order to be able to suggest important criteria, to bootstrap the process if necessary.

Steps The chauffeur uses the tool to display the state of the context. There are two different phases:

1. If there exist mandatory rules, they are "law" and cannot be questioned. The group analyzes the properties of the candidates automatically sorted by these rules, in order to build up references for further discussions.
2. The group iterates through steps 2a to 2f until either all candidates have been sorted and participants are convinced that it is fair enough, or participants cannot find any more criteria on which to discriminate on a consensual way, or time is out.
   (a) If there are revisable rules, the group investigates what their impacts on the given context. These rules come most likely from a previous meeting of the same kind. They are not necessarily totally relevant for the current context. The group decides to keep them, adjust them or leave them aside.
   (b) The group takes the list of criteria and decides which ones are relevant for the decision. Note that not all criteria need to be investigated in depth. For a jury the group can decide that the grade for each particular module does not need to be investigated for the time being. Nothing prevents the group from coming to that point later.
   (c) When a set of criteria have been accepted as being relevant, the group discusses what the characteristic values for these criteria are for each category.
   (d) Whenever a logical formula has been identified, a rule can be created to keep a trace of each small decision. It provides basic blocks for the global explanation of the final decision. It also enables the group to question each of the small decisions at anytime during the meeting.
   (e) The group can decide, at any time, that new criteria are relevant. If these criteria were not initially in the context, their values can be filled in on the fly.
   (f) Regularly, the rules are inspected to check that they still reflect the current state of the group's understanding and consensus.

Tool and Manpower

As reported in (Ducassé and Ferré, 2008), spreadsheet can be used to support the LogicalMulticriteriaSort thinkLet. Their filters, masks and macros provide part of the necessary functionalities. However, selecting criteria and candidates in the spreadsheet is error prone. It is hard to ensure consistency. Furthermore, adding attributes in the table is tedious and again error prone. Keeping track of the selection process is almost impossible. LIS tools, as shown in the Test Case Section 4, are appropriate tools to sustain this thinkLet. A chauffeur is necessary for a physical meeting, a facilitator is also required. Unless for simple cases, it is advisable that the roles are played by two different persons.

Insights on LogicalMulticriteriaSort

LogicalMulticriteriaSort is related to two thinkLets of (Briggs and de Vreede, 2009): ***ChauffeurSort*** and ***Multicriteria***. ***ChauffeurSort*** investigates the candidates in sequence. Briggs and de Vreede state "*do not use this thinkLet if time is of the essence*". Consistently, (Lewis, 2010) states "*often, the majority of the time will be spent on the first few ideas [candidates] at the top of the list, whether these have the greatest merit or not*". With LogicalMulticriteriaSort it is the criteria that are investigated following an order prompted by the participants. It ensures both fairness and speed. Going along the candidates without consistency checking mechanisms, the treatment of candidates is not fair, for one candidate participants will talk about some criteria, for another candidate there is no guarantee that the same criteria will be used. With LogicalMulticriteriaSort, even if all the criteria have not been considered, it is guaranteed that all the candidates will have been considered along the discussed criteria. As a consequence, the meeting can stop after any criterion analysis, preferably if the group decides that enough relevant criteria have been considered. If time is out, the criterion analysis is incomplete, the decision is nevertheless guaranteed to be fair to the candidates.

The ***Multicriteria*** thinkLet is based on Multi-Criteria Decision Analysis (see for example (Zopounidis and Pardalos, 2010)). Criteria are given a numerical value and a weight, and then a weighted sum is computed. Briggs and de Vreede state "*do not use this thinkLet as final decision-making process. Odd anomalies can crop up in the results of Multicriteria analysis*". Consistently, (Bana e Costa and Chagas, 2004) emphasize that producing numerical criteria weights is tedious, arbitrary and can lead to dubious results. In order to palliate those problems, they ask users to fully rank the criteria, then according to the actual values of the data they automatically produce weights to insure consistency in the actual context. While this is a significant improvement, our experience is that it is often difficult to reach an agreement about a total ranking of the criteria at a meeting. For example, in the test case of Section 4 there was no obvious convergence in the relative importance of the criteria before the meeting. Even at the end of the meeting there was still disagreement about the importance of a few criteria. With our approach, the values can be numerical but also symbolic, their analysis is logical. Participants only have to decide if a criteria is relevant or not and put conditions, for example thresholds, on the relevant criteria. When the results of LogicalMulticriteriaSort lead to a decision, they explicit in a legible way arguments on which the group agrees, and can be relied upon. When the results do not lead to the final decision they still pave the way for further steps. They, thus, help the group endorse the decision.

4 LogicalMulticriteriaSort Success Stories

An example of success story is a recruitment process, reported in (Ducassé and Ferré, 2008), whose first step had followed the LogicalMulticriteriaSort thinkLet, even if not explicitly said. The overall objective of the meeting was to propose a sorted list of candidates. The first step consisted in sorting the candidates into three categories, "to be considered", "may be", "excluded". For the further steps only the candidates in the first category were actually considered. In the remaining of this section, we describe a successful test case related to a jury at a technical university.

Context of the Jury Test Case

The reported test case is a debriefing of a year validation jury (called the *actual jury* in the following) whose results had been controversial. Such juries, seat at the end of every school year and decide for each student of a class whether s-he passes or fails. In the past, the juries of the concerned institution use to reason almost exclusively on a global weighted average calculated over all the grades of the student for the school year. With the European Bologna process[2], ECTS credits came into the picture. The students must acquire 30 ECTS credits per semester. Yet, compensation between modules is allowed in the French

[2] http://www.europe-education-formation.fr/bologne-ects-doc.php

system[3] and the institution introduced compensation rules for automatic year validation. Jury members are therefore forced to reason on multiple criteria, at least on credits and global average.

The authors were respectively the facilitator and the chauffeur of the meeting. Camelis was used (see Section 2). Beside the facilitator and the chauffeur, there were five participants, all computer science teachers not familiar with the LIS tools. Among the five participants, 4 attended the actual jury; the remaining person had chaired another jury at the same period. Neither the facilitator, nor the chauffeur had attended the actual jury. The whole actual jury, 8 members, had been invited but only the people with responsibilities in the institution came, namely all participants were responsible of a curriculum, including the chair of the actual jury. There were no identified conflicts but there was also no obvious convergence in the relative importance of the criteria. The discussions during the meeting were (audio) recorded.

A spreadsheet file had been prepared for the actual jury by the chair of that jury. It contained 55 lines (one per student under judgment) and 160 columns (one per criteria). Examples of criteria are the grading and the acquired ECTS credits of each module, the average per group of modules, which specialty students took or the ranking in the class. A printed version of this file had been used at the actual jury. The jury chair had sorted it into several sheets. The actual jury members had to browse through 9 printed pages. The facilitator transferred the spreadsheet data into Camelis and structured by hand the criteria so that participants did not have to face the 160 criteria in a first step. Figure 1 shows what participants could see during the meeting (explanations are given in the next section).

Meeting First Phase: Analysis of Automatically Passing Students

The first phase, analyzing the impacts of the mandatory rules, lasted approximately 20 minutes. There was only one rule specifying the students who were automatically passing thanks to the institutional compensation rules. Figure 1 shows a screen shot of Camelis during the first phase. LIS user interfaces give a local view of the concept lattice. The local view is made of three parts: (1) the *query* (top left), (2) the *extent* (bottom right), and (3) the *index* (bottom left). The *query* is a logical formula that typically combines criteria (e.g., `g1_Average`), patterns (e.g., `g1_Average >= 10.`), and Boolean connectors (`and`, `or`, `not`). On the figure, the query area shows the implementation of the institutional automatic passing rule: `g1_Average >= 10. and g2_Average >= 10. and g3_Average >= 10. and g4_Average >= 10. and g5_Average >= 10.` . It means that the selected (passing) students have a grading average of at least 10 (out of 20) for all of the 5 groups of modules.

The *extent* is the set of objects that are matched by the query, according to logical subsumption. Objects correspond to the candidates mentioned in the description of the LogicalMulticriteriaSort thinkLet. The candidates are actually students in this test case. On the figure, one can see part of the identifiers corresponding to the 44 students passing thanks to the institutional automatic passing rule. Note that during the test case participants could see the full name of students.

Finally, the *index* is a set of criteria, taken from a finite subset of the logic, it is restricted to criteria associated to at least one object (student) in the extent. The index plays the role of a summary or inventory of the extent, showing which kinds of objects there are, and how many of each kind there are (e.g., in Figure 1, 3 students in the extent have 10 `compensated_credits`.)

Note that the query had been obtained solely by clicking on criteria of the index. Let us describe how it had been produced. Firstly, opening the `Average ?` criterion, the chauffeur could click to open `g1_Average`. Then clicking on one of the displayed values (here `10.`), then on the `>=` button and then on the `zoom` button produced the `g1_Average >= 10.` part of the query. Repeating the process for all the group averages produced the query.

[3] See for example "Arrêté du 1er août 2011 relatif à la licence NOR: ESRS1119411A"

1.

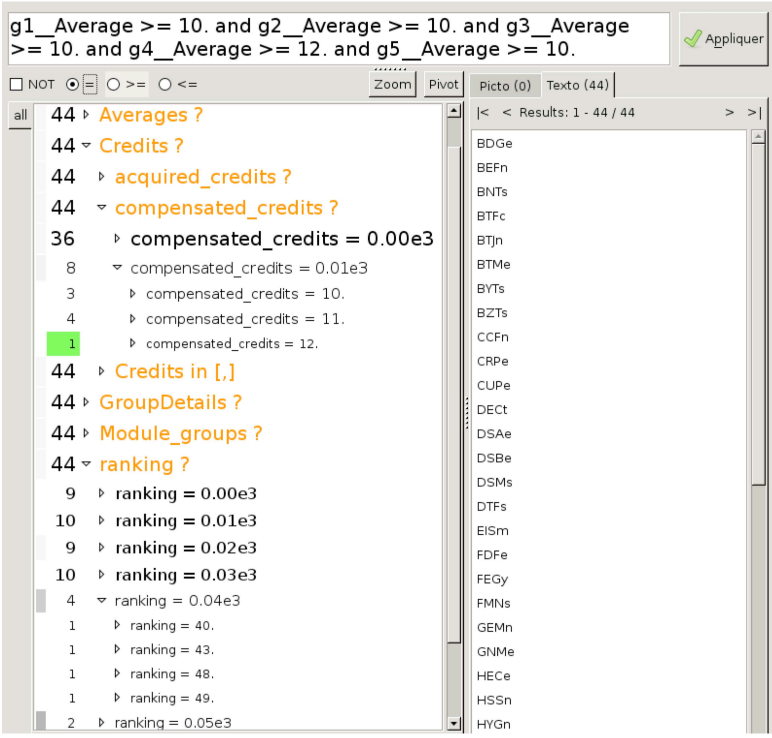

Fig. 1 Screen shot of Camelis with a query specifying a mandatory rule

After some investigations, the group agreed that two interesting facts about the passing students disserved to be noted. Firstly, as can be seen on the figure, 8 out of the 44 passing students had 10 or more `compensated_credits`. Compensated credits come from the institutional compensation rules. The students failed some modules, but because they had grades good enough in some others they gain credits for modules for which they fail. It was very important for the following discussions to note that the maximum number of compensated credits was 12. Secondly, it can also be seen that there are some "holes" in the ranking. `ranking = 0.00e3`[4] means that there are ranking values between 0 and 9, `ranking = 0.01e3` means that there are ranking values between 10 and 19, etc. As mentioned above, the number in front of each criterion tells how many students have the criterion. Here it seems normal that there are 9 students ranked between 1 and 9. It is less normal that there are only 9 students ranked between 20 and 29, and only 4 between 40 and 49 especially as there are two automatically passing students who were ranked after 50. The global ranking, based on the global grade average, used to be the main decision criteria. Here we can immediately see that at least 7 students fall into an unusual case.

Meeting Second Phase: Sorting Out Candidates

The second phase, the actual sorting phase, lasted approximately 50 minutes. In the meeting there were no prior optional rules. To initiate the discussion the facilitator suggested that the number of credits was probably a relevant criterion. After a discussion, the group hinted that it would be unfair to require more credits from the students under discussion than the maximum number acquired without compensation by the automatically passing students, namely 48 credits. Figure 2 shows a screen shot of Camelis where the query selects the students who do not automatically pass and who have acquired at least the required 48 credits. In the extent area one can see that 5 students are concerned. In the index, one can see a number of interesting points. Firstly, all concerned students have a general average above 11 (2 even have an average above 12). They also all have a scientific average above 10. The three students who are below 10

[4] We acknowledge that this is off putting and we are investigating how to present this in a more accessible way. Note, however, that participants, although initially puzzled, managed very well after they received explanations.

for the average of the g1 group of modules (e.g. `g1_average = 0.00e3`) are the same three students who are above 10 for the average of the g3 group of modules (e.g. `g3_average = 0.01e3`). It can be seen from the coloring of the numbers, an identical color means that the related students are the same. From the above properties, the group decided that for that particular class, it was ok to let through those students. The chauffeur therefore created a rule, saved in the tool, to keep track of the reasons for which these students were let through. At the moment of editing the rule, somebody said that the rule was only acceptable for him because the scientific average was not too weak and wanted that to be recorded in the rule. The rest of the group agreed and the actually saved rule is given in Figure 3. It uses another rule (not shown here) that specifies that if the scientific average is below 10, it is considered too weak and the student fails. Note that the 5 concerned students automatically got a new criterion, namely 'let through cause more acquired credits than some automatically pass students'. This new criterion could then be used in queries. The meeting went on by sorting either from top or bottom. Sorting from top consisted in identifying thresholds above which students could pass (for example with a general average above 12), sorting from bottom consisted in identifying thresholds below which students should fail (for example with not enough acquired credits or a scientific average too weak).

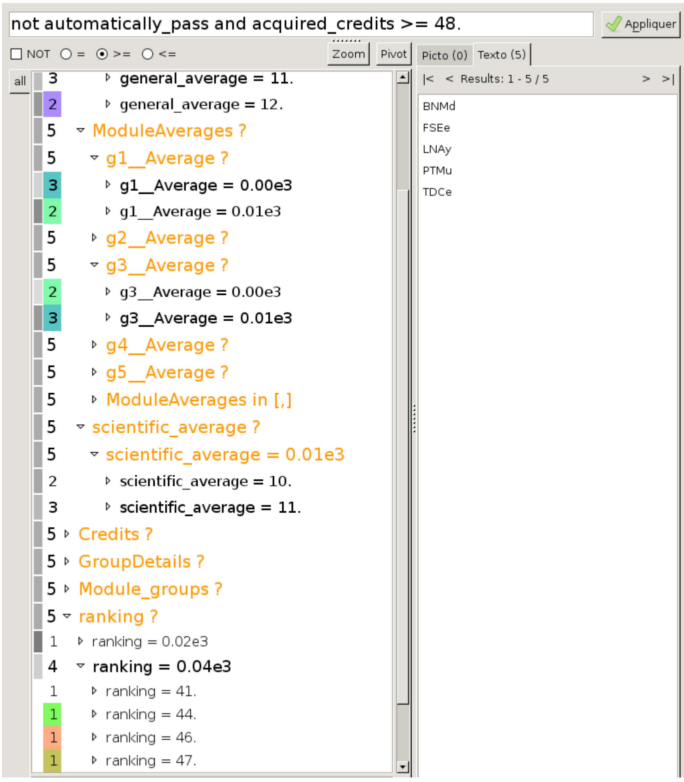

Fig. 2 Screen shot of Camelis when the group had identified a rule

```
'let through cause more acquired credits than some automatically pass
students' :-
           not automatically_pass
     and not 'fail due to scientific average too weak'
     and acquired_credits >= 48.
```

Fig. 3 First rule produced by the jury test case

At the end of the meeting, the context had been enriched by 5 rules which sorted 53 out of the 55 students: (r1) forty four students automatically passed thanks to the institutional mandatory rule, (r2) five students were let through thanks to the rule discussed above, (r3) one student was let through thanks to a good enough general average, (r4) one student failed due to a scientific average too weak, and (r5) 3 students failed due to a lack of credits. One student was concerned by two (consistent) rules and he is therefore counted for each of the rules. Altogether the five produced rules use the three logical connectors "and", "or" and "not". They also use 6 different criteria, symbolic and numeric, showing that multi-criteria reasoning and decision are indeed possible at a logical level. Rule r3 concerns one student only. There was a consensus that the student should pass. The facilitator asked whether the student should be simply moved to a "pass" basket. One participant asked that a rule was created to specify explicitly why participants thought he should pass and to keep a trace of the reasons.

At the end of the meeting, there were only two students for whom no consensus could be found, whereas participants reported that the actual jury voted for 6 students.

Users' Feedbacks

This section reports the results of the two questionnaires filled by participants, one before and one after the meeting. Note that the person who did not attend the actual jury answered only the general questions. The questionnaire and discussions were in French. The quotations of participants have been translated by the authors.

Questionnaires Before the Meeting. All participants reported that at the actual jury there had been no formal step to analyze the results of the students who automatically passed. Two participants mentioned brief discussions about some students while discussing the other students. Three participants acknowledged that analyzing these results with the spreadsheet display is too tedious. The arguments were that the data were too numerous, too complicated and that it takes too much time. One participant thought that reading the 9 pages of spreadsheets was not a problem. Four participants reported having clearly conscience that they were doing a multicriteria decision at the actual jury. Three, out of the four persons who answered, reported being rather unsatisfied of the decision of the actual jury, whereas the 4th one was rather satisfied. Stated reasons for unsatisfaction were: "*we have been unfair against one of the student*", "*some important arguments had not been explicitly said*". The two participants who had to deal with the students afterwards were among the unsatisfied participants and reported having difficulties to endorse the decision.

Questionnaires After the Meeting. The two participants who had to endorse the decision reported that after the meeting they endorse it better, even if they were still unsatisfied with the decision. The four participants who answered the question reported that the first phase (analyzing the students that automatically pass) had been useful: "*it has put the light to the importance of the compensated credits*", "*it has made me realize that we do that but informally*". All participants validated the produced rules: "*I realize now that they are exactly the ones we used, even if I was not totally aware of it last time*", "*they could be used at the next jury as a discussion basis*", "*they will help guarantee consistency in the jury decision*", "*with the rules it is more rational, the memory effect is interesting*", "*I appreciate to have clear and stable rules*". Four participants agreed that the tool had helped the group to express the rules. The person who said having no problem to analyze the printed spreadsheet pages thought that it was rather the opposite. All participants reported having understood all the queries and their effect. Four participants considered that they contributed to the queries (note that a rule is simply a given query which has been given a name). One of them emphasized that it has been a collective contribution. The participants agreed to use the tool for the forthcoming jury of the same level together with the usual material. It should be noted that the two persons who had to endorse the decision were the most positive about the results of the meeting. Moreover, the chair of the actual jury, who was one of those two persons, was the most positive of all.

5 Conclusion

In this article we have proposed the LogicalMulticriteriaSort thinkLet. To address the problems of a multicriteria decision LogicalMulticriteriaSort considers criteria one at the time as the take-the-best heuristics. There is no predefined order between criteria, participants put forward the ones they find the most relevant at a given time. The values can be numerical or symbolic; the analysis on them is logical. It avoids the need to give artificial values and weights to the criteria as opposed to the Multicriteria thinkLet. LogicalMulticriteriaSort guarantees more fairness and speed than the ChauffeurSort thinkLet. In addition, LogicalMulticriteriaSort is supported by a GDSS, based on Logical Information Systems. Thanks to the tool, the group can share a better understanding of the situation and it can be easier to collectively endorse a sensitive decision. In particular the tool keeps tracks of all of the decisions taken so far with a set of rules that explain how candidates have been sorted. It also gives an instantaneous feedback of each current decision. LogicalMulticriteriaSort has been tested on the debriefing of an academic year validation jury whose results had been controversial. The test case participants were positive about the process and they all agreed to use LogicalMulticriteriaSort and the supporting tool for the forthcoming jury at the same level. The two persons who had to endorse the decision were the most positive about the results of the meeting.

Acknowledgements The authors thank their colleagues from the computer science department of the INSA of Rennes who attended the "jury" test case. They thank Sébastien Ferré for the thorough technical support on Camelis. They thank Alice Hermann for her help on the design of the questionnaires.

References

Adla, A., Zaraté, P. and Soubie, J.-L. (2011). A Proposal of Toolkit for GDSS Facilitators. Group Decision and Negotiation, 20, 57–77. 10.1007/s10726-010-9204-8.

Bana e Costa, C. A., and Chagas, M. P. (2004). A career choice problem: An example of how to use MACBETH to build a quantitative value model based on qualitative value judgments. European Journal of Operational Research, 153(2), 323–331.

Briggs, R., and de Vreede, G.-J. (2009). ThinkLets: Building Blocks for Concerted Collaboration. Center for Collaboration Science, University of Nebraska at Omaha, USA.

Briggs, R. O., Kolfschoten, G. L., de Vreede, G.-J., Albrecht, C. C., and Lukosch, S. G. (2010). Facilitator in a Box: Computer Assisted Collaboration Engineering and Process Support Systems for Rapid Development of Collaborative Applications for High-Value Tasks. Pages 1–10 of: HICSS. IEEE Computer Society.

Davis, A., de Vreede, G.-J., and Briggs, R. (2007). Designing ThinkLets for Convergence. In: AMCIS 2007 Proceedings.

Ducassé, M., and Ferré, S.. (2008). Fair(er) and (almost) serene committee meetings with Logical and Formal Concept Analysis. In: Eklund, Peter, and Haemmerlé, Ollivier (eds), Proceedings of the International Conference on Conceptual Structures. Springer-Verlag. Lecture Notes in Artificial Intelligence 5113.

Ferré, S., and Ridoux, O. (2000). A Logical Generalization of Formal Concept Analysis. Pages 371–384 of: Mineau, G., and Ganter, B.(eds), International Conference on Conceptual Structures. Lecture Notes in Computer Science, no. 1867. Springer.

Ferré, S., and Ridoux, O. (2004). An Introduction to Logical Information Systems. Information Processing and Management, 40(3), 383–419.

Ganter, B., and Wille, R. (1999). Formal Concept Analysis: Mathematical Foundations. Heidelberg: Springer.

Hiltunen, V., Kurttila, M., Leskinen, P., Pasanen, K., and Pyklinen, J. (2009.) Mesta: An internet-based decision-support application for participatory strategic-level natural resources planning. Forest Policy and Economics, 11, 1–9.

Kilgour, D. M., and Eden, C. (2010). Handbook of Group Decision and Negotiation. Advances in Group Decision and Negotiation, vol. 4. Springer Netherlands.

Kolfschoten, G. L., de Vreede, G.-J., and Briggs, R. O. (2010). Collaboration Engineering. In: (Kilgour and Eden, 2010).

Lewis, L. F. (2010). Group Support Systems: Overview and Guided Tour. In: (Kilgour and Eden, 2010).

Marewski, J. N., Gaissmaier, W. and Gigerenzer, G. (2010). Good judgments do not require complex cognition. Cognitive Processing, 11(2), 103–121.

Shanteau, J. (1992). How much information does an expert use? Is it relevant? Acta Psychologica, 81(1), 75–86.

Tilley, T. (2004). Tool Support for FCA. In: Second International Conference on Formal Concept Analysis. Lecture Notes in Computer Science, vol. 2961. Springer.

Vogel, D., and Coombes, J. (2010.) The Effect Of Structure On Convergence Activities Using Group Support Systems. In: (Kilgour and Eden, 2010).

Zopounidis, C., and Pardalos, P. M. (2010). Handbook of Multicriteria Analysis. Applied Optimization, vol. 103. Springer Berlin Heidelberg.